Network Simulator - 2021

Overview

The software consists of graphical user interface (GUI), which is built on top of the Qt framework. Through the GUI, the user can create different network scenarios by adding nodes and the links between them. After the simulation is finished, a summary of the results can be displayed in the console

This project only considers two types of nodes – end-hosts and routers, hence it is not possible to have a switch. Currently it is possible to have only one network setup at the time. Also, the simulation is only running one scenario at a time.

Functionalities and features

The design comprises of two parts, the front-end and the backend. The frontend deals with the design of the user interface. The interface has a window from which nodes can be selected. The nodes can be dragged and dropped into the main window where the network scenarios are built. The link between the nodes can be added by simply clicking once on the link icon and then drawing a line between two nodes. A scenario can be added by clicking once on the packet icon and clicking once on each of the two PCs nodes.

After the set-up is done, the simulation is run with a click on the 'Run' button. The progress of the simulation can be viewed in the GUI terminal and the summary will also be printed out once finished.

The backend part of the software is where the implementations of the header files and the source files (C++ classes) are kept. These classes provide the functionalities needed to run the front-end when a simulation instance is created. The functionalities and further explained below.

Adding nodes: The simulator includes two different nodes: *routers* and *end-hosts*. Routers can only forward data and end-hosts can send and/or receive data. User can place the nodes wherever wanted in the given space. Currently placeholder names are also generated by the system. Router default queue size is 100.

Adding links: Once a user has added at least two nodes, the user can add a link between them. Transmission speed unit is Kbps in our system and currently its default value is 100.

Running a packet flow scenario: Simulator can run one scenario at a time. Configuring a scenario, a user needs to set the source node that will send the data, the destination node that will receive the data by selectin the packet object once and clicking on top of the PC node. The data size is set as default 1 MB. After the nodes and links are set, the user can run the simulation. The system splits the data into smaller network packets. For simplicity we have chosen to use Ethernet MTU for the packets which are 1500 bytes.

Follow the progress of simulation: When the simulator is computing the scenarios, the progress will be printed out in text form to the GUI console.

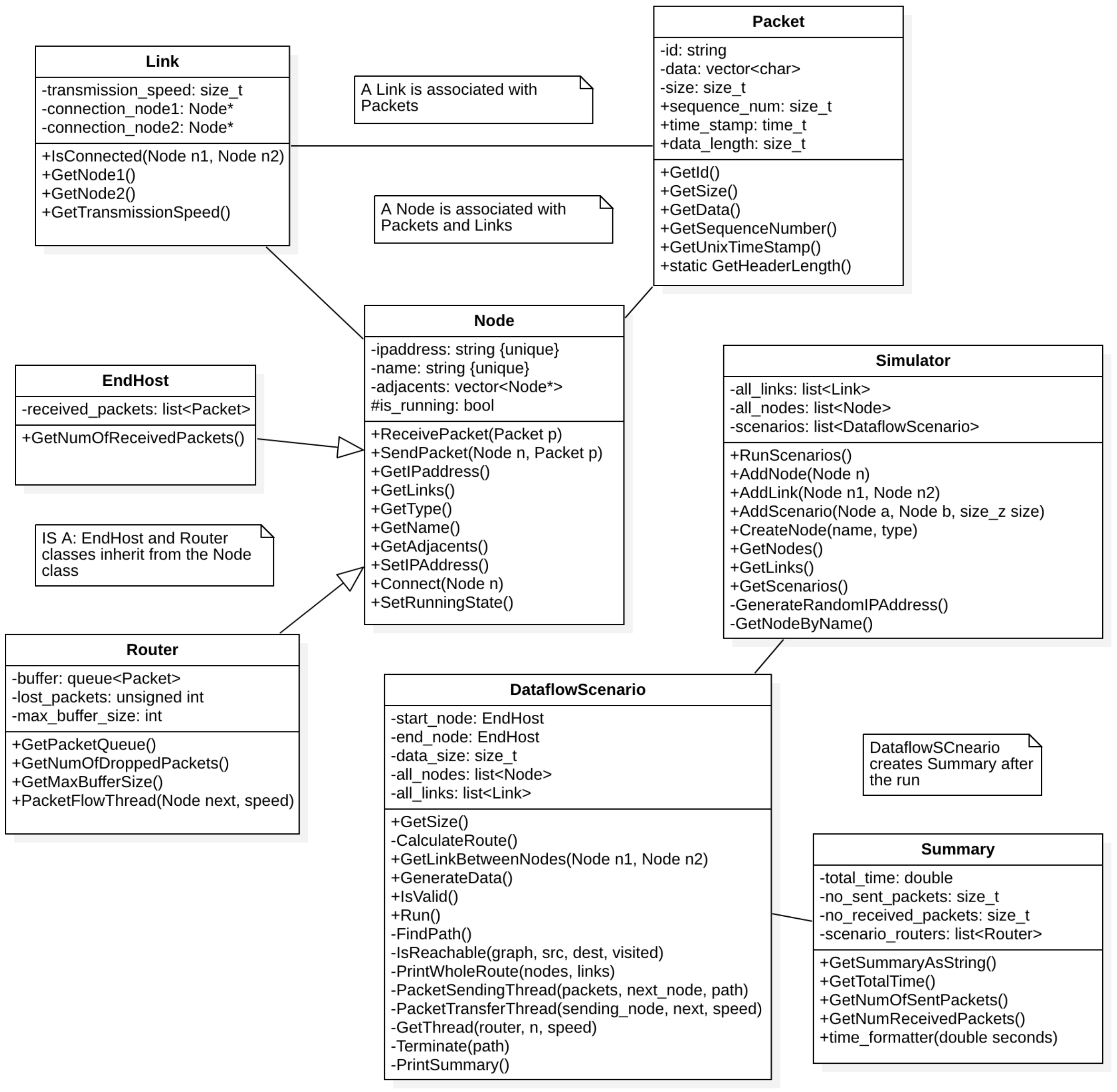
See summary of the simulation: Once the simulation is finished, summary will be displayed to the user. The simulator collects the following: total time, average time/packet, number of sent packets, number of received packets, and number of dropped packets/router.

Software structure

The UML below presents the class hierarchy of the backend classes. *Simulator* class is a central piece that takes care of setting the network elements. It allocates the IP addresses and takes care that right Nodes get connected with a Link object. *Node* is an abstract class that can be either *EndHost* or *Router* in our system. *Link* class represents a connection between nodes. *Packet* represents a network packet that gets sent between nodes.

DataflowScenario class takes care of running the simulation i.e, scenario. To find the path between the two end hosts, it uses DFS algorithm. If the path is invalid, the simulation will not run. DataflowScenario will split the bigger data piece to packets before the run according to MTU.

To better simulate real life networks, the system uses multithreading for the packet flow implementation to have little bit like TCP socket behavior. Each router and the sending node in a path operates in its own thread and the packets will be sent according to the link speed between the nodes.



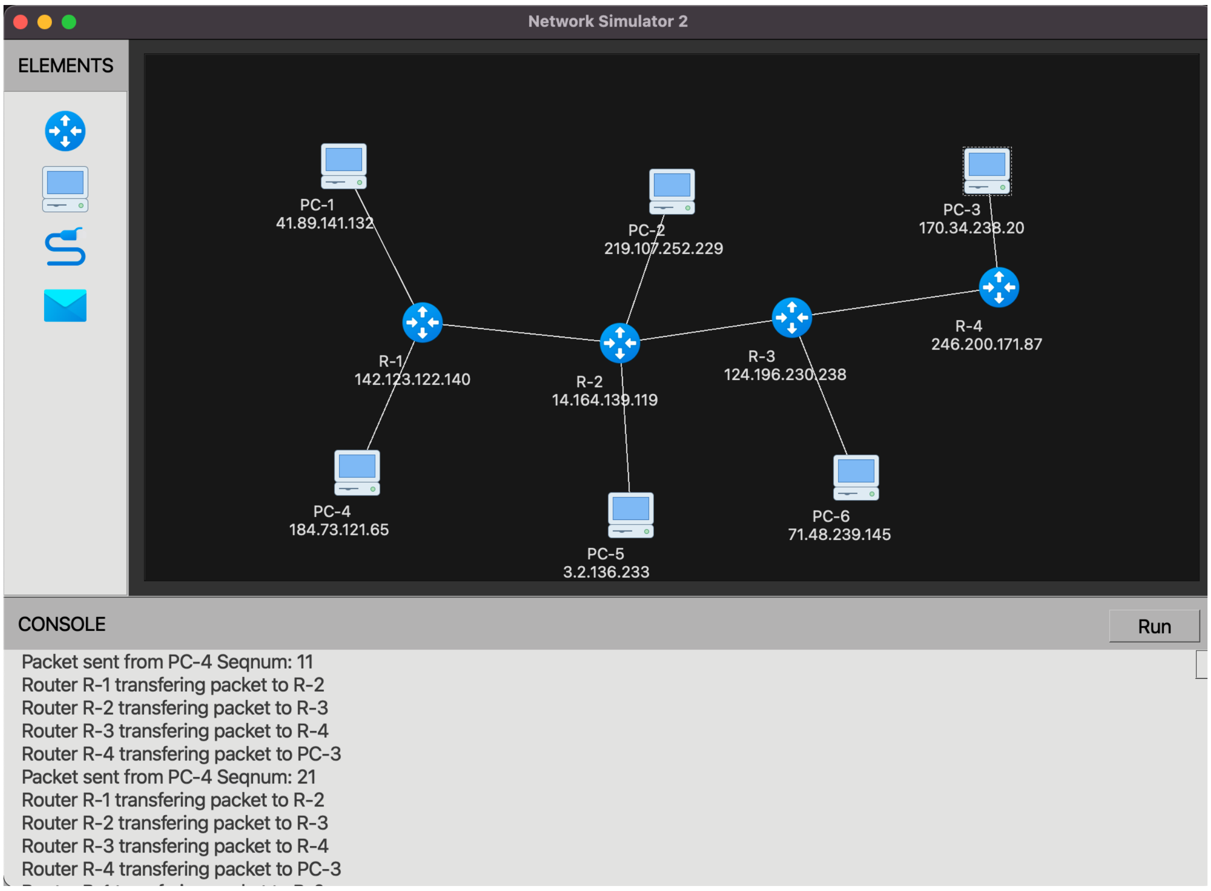
Instructions for building and using the software

Qt libraries need to be installed. Look more from lib folder.

Building with Makefile:

- (make clean)
- make build
- make run

If using MacOs you might have ro manually save the src.pro file (command + s). Once project runs successfully, the GUI will open (See Figure 1). Select nodes from sidebar and drag them to the canvas. When adding Links, click the link icon once and then drag a line between nodes. When configuring dataflow scenario, click the packet object and select the end nodes that you want to transfer packets between. Once done, you should be able to click the Run button and you can follow the simulation from console.



```
Packet sent from PC-4 Seqnum: 11
Router R-1 transfering packet to R-2
Router R-2 transfering packet to R-3
Router R-3 transfering packet to R-4
Router R-4 transfering packet to PC-3
Packet sent from PC-4 Seqnum: 21
Router R-1 transfering packet to R-2
Router R-2 transfering packet to R-3
Router R-3 transfering packet to R-4
Router R-4 transfering packet to PC-3
```

Figure 1. Simulator GUI

Testing

The project includes some simple test suites. Due to time resources, not too many tests were made. Currently there are testing issue due to back-end and front-end integration as the test suite cannot recognize Qt components. To fix this issue, the *NsTerminal* should be removed from backend. However, the tests were still used through the whole project until almost the very end.

Test with Make:

- make build
- make test

Test of Simulator

- AddNode: Adds EndHost and Router objects to the system
- AddLink: Adds Link objects (connections between nodes) to the system

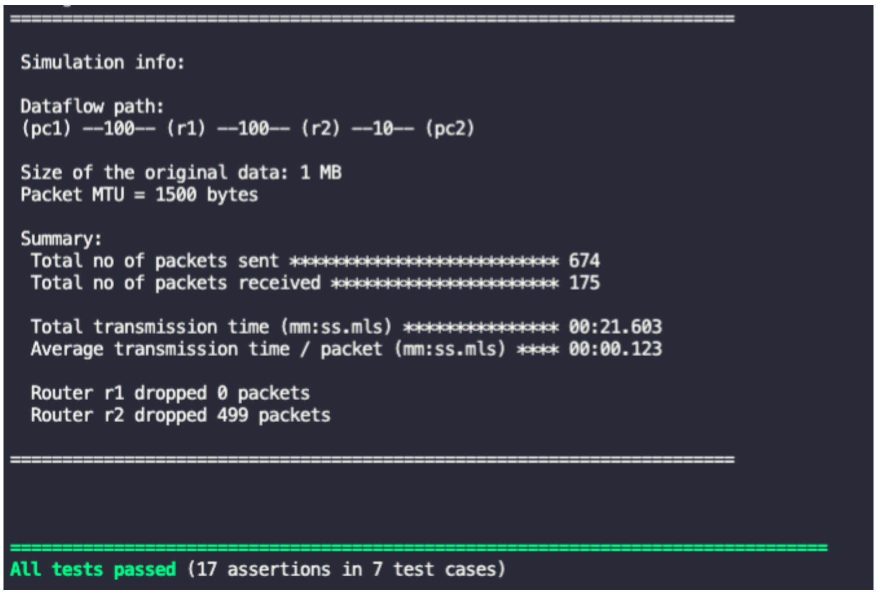
- AddScenario: Adds Dataflowscenario object to the system, this does not check that the scenario is valid! Only that is has been successfully added

Tests of Node

- Only operator== method tested in this suite. However, this method was used a lot in a project.

Test of DataflowScenario

- New Instance: Simple init tests and testing with invalid path.
- Run: One end-to-end test for utility purposes to run the scenario, super helpful (See Figure 2).



```
===============================================================================

 Simulation info:

 Dataflow path:
 (pc1) --100-- (r1) --100-- (r2) --10-- (pc2)

 Size of the original data: 1 MB
 Packet MTU = 1500 bytes

 Summary:
  Total no of packets sent ************************** 674
  Total no of packets received ********************** 175

  Total transmission time (mm:ss.mls) *************** 00:21.603
  Average transmission time / packet (mm:ss.mls) **** 00:00.123

  Router r1 dropped 0 packets
  Router r2 dropped 499 packets

===============================================================================



===============================================================================
All tests passed (17 assertions in 7 test cases)
```

Figure 2. Test results

Work log

Description of the division of work and everyone's responsibilities. You can find more from the Meeting-nodes.md.

Prince:
Responsible for front-end implementation and backend integration of Qt framework libraries

Milla:
Originally was supposed to work on the GUI side but ended up building the backend side. Responsible for putting the backend classes together to build the packet flow system end-to-end. Most part of the documentation.

Hechukwedere:
Responsible for project configuration set-up, GUI implementation, and backend integration.

Oriyomi:
Responsible for the back-end implementation. I only worked on a few classes.

Hours/week

	44	**45**	**46**	**47**	**48**	**49**	**50**	**Total**
Prince	20	20	15	15	15	15	30	130
Milla	10	15	20	10	15	25	30	125
Hechukwedere	10	10	25	18	20	20	25	198
Oriyomi	15	15	12	10	16	18	25	111